An Apache Based, Intelligent IoT Stack

Trevor Grant

About Me

Trevor Grant

PMC Apache Mahout Project

PPMC Apache Streams-Incubator

Open Source Evangelist, IBM

@rawkintrevo

rawkintrevo@apache.org

http://rawkintrevo.org

Huge shout out to **Joe Olson**, couldn't be here today but did all the hard stuff.

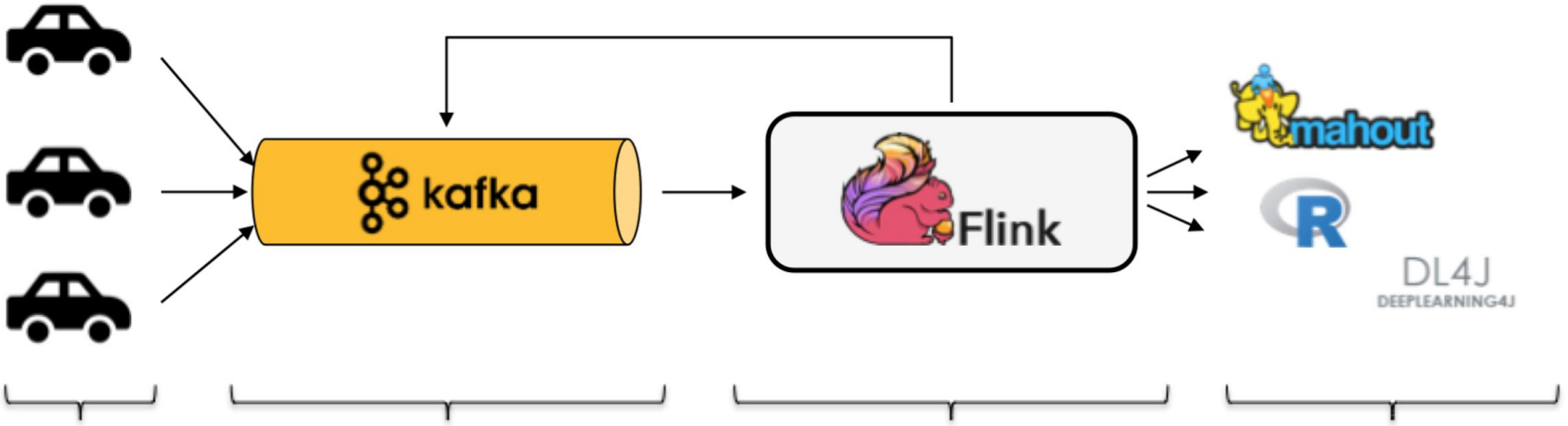
4 Stages of Any IoT App

- Collecting
- Storing
- Processing
- Analyzing

Different projects / approaches to *where* this happens (edge / central)

Maybe multiple places (some stuff local, some stuff pushed to server)

The “Stack”


kafka
Flink
mahout
R
DL4J
DEEPLEARNING4J
Collect
(MC3)
Store
(Kafka)
Process
(Flink)
Analyze
(Mahout)

Device Level (Collect)

Vehicle Sensors

Various vehicle sensors in modern vehicles:

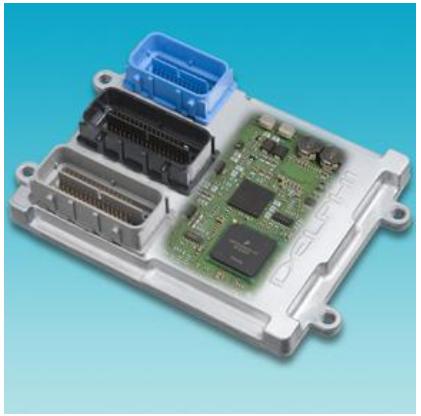
1. Temperature
2. Humidity
3. Liquid level
4. Mass air flow
5. Pressure
6. Position
7. Fluid property
8. Vehicle location

Sensors interface with the rest of the vehicle via the vehicle data bus

Engine Control Module

- Combine sensor data to control engine performance
  - Air / fuel ratio
  - Idle RPM
  - Variable valve / electronic valve timing

- Can be programmable
- Stores fault codes
- Interacts with the rest of the vehicle via the vehicle data bus



Delphi MT88 ECM

Morey MC3

- Telematics device used in vehicle to get data from devices on the vehicle bus (sensors and ECM)
- Uses the cellular network to move data off the vehicle and into a data center for processing
- Plugs into vehicle bus using standard interface, so it works in most vehicles (personal, commercial, industrial)
- Customizable based on application and sensors available
- Works with standard data buses: CAN, GMLAN, J1850, J1708, KWP2000

Apache Alternative

Morey MC3 Costs $225 each on Amazon
http://www.ebay.com/itm/Telematics-GPS-Fleet-Management-Device-Morey-Corp-Hawk-MC-3-CDMA/252015674910

Alternative- Raspberry Pi (or similar) with Apache Edgent-incubating

Cheaper- more work on front end.

ASFv2 Licensed projects exist for reading ODB2, no specific project.

Reading Raw

- All vehicle data reported from the MC-3 is in binary message format
- Message types for various events
  - Location report
  - Motion start
  - Geofencing
  - Hard braking
  - User defined
  - Trouble codes
- Each message type has its own structure that needs to be converted into something that can be processed.
- C library provided by vendor

JSON Output


```
{"DeviceType": "WFT_VTS", "Message": {"gps_based_altitude": 233, "longitude": -88.0332,
"packet_timestamp": "2017-04-29 16:59:24", "latitude": 42.0421, "device_message_ctr": 1177,
"daq_timestamp": "2017-04-30 19:38:21", "distance_travelled_odometer_using_gps": 18610,
"relative_signal_strength_from_modem": 16, "coolant_temperature": 112,
"vehicle_speed_gps_accelerometer": 65535.0, "engine_rpm": 821, "null": 0,
"message_event_id": "LOCATION", "gps_based_heading": 0, "gps_based_speed": 0,
"gps_number_of_satellites": 7, "gps_pdop_value": 22, "gps_vdop_value": 19,
"gps_hdop_value": 11, "engine_status": 0, "battery_voltage_millivolt": 14736,
"vehicle_motion_status": 0, "gps_fix_validty_falgs": 1, "vts_device_id": 45317471817796386,
"fuel_level_1_from_vcm": 26, "fuel_level_2_from_vcm": 255}, "InstanceId": "1", "DeviceId":
"45317471817796386", "ReceivedTimeStamp": "2017-04-30 19:38:21", "SequenceId": "0000",
"MessageId": "LOCATION"}
```

Stream Processing (Store / Process)

Apache Kafka

- Many options for storing data
  - Relational, NoSQL, raw files, timeseries DBs…….
- What format? JSON? Binary?
  - Depends on how you want to store it.
  - Depends on how frequently you access it - serializing and deserializing can be expensive
- Are you going to need to reprocess some / all of it?
- What processing engine(s) need to access the data? (Connectivity)
- How scalable? Fault tolerant? Cost?
- Apache Kafka has out of the box functionality to address all of these issues.

Apache Flink

- Many options for processing the data
- Treat data as a stream (vs batch processing)

Why do we need Kafka

This is a toy/POC- for more Complex Event Processing need Flink

“A.I.”
a.k.a.
Analyze It

Apache Mahout

Why Mahout?

- Is Apache. (this is a an Apache-All-The-Way stack)
- Most sophisticated and diverse ML in Apache Ecosystem*
- *Native Solvers*- can optimize incore BLAS operations on ANY architecture
- Models can be trained on distributed datasets then pushed down to edge device

Methods we care about here:

- Correlated Co-Occurrence (CCO) Recommender
- MLP (well- sort of-almost)

CCO : Overview

Consider a “primary action matrix” - Rows are “users” or “vehicles”

Columns might be:

- Purchased Item (eCommerce)
- Maintenance Triage (Automotive)
  - Critical (Shut Engine Down Now)
  - Urgent (Pull off at next exit, call for tow)
  - Urgent - User1 (Pull off at next exit, check tire pressure/oil/coolant)
  - Priority (Deadline vehicle at end of trip for shop maintenance)
  - Convenience (Something is off- get to the shop soon)
  - Routine (You’re due for a trip to the shop)

CCO : Overview

Consider a “secondary action matrices” - Rows are “users” or “vehicles”

Columns might be:

- Viewed Item (eCommerce)
- Sensor Status (Automotive)
- Mileage (Automotive)
- ODB2 Error Codes (Automotive)
- Driver technical ability (some drivers have more maintenance training)(Auto)

CCO : Overview (Math)

Consider primary matrix A

Secondary Matrices B, C, D, …

User has history vector h_a, h_b, h_c, … vector positions correspond to columns

User Recco = $LLR(A^TA) \cdot h_a + LLR(A^TB) \cdot h_b + LLR(A^TC) \cdot h_c + ...$

LLR = Log Likelihood Ratio - test that items are not related (lower is better)

Mahout uses LLR inverse, bigger is better.

When to Use CCO

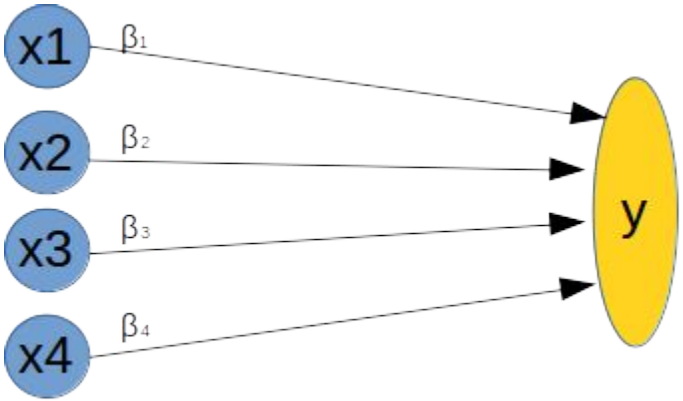
To be fair-
Mahout has AWESOME recommenders and I'm admittedly biased

When to Use CCO

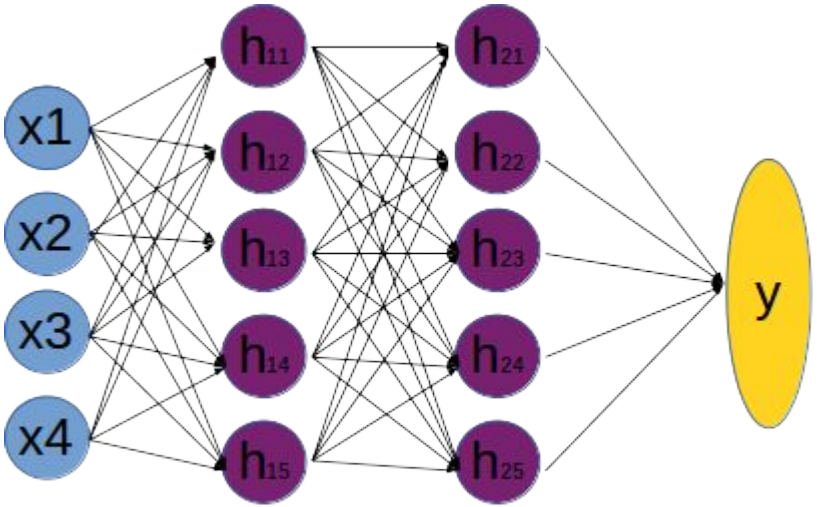
- Computationally Cheap
- Easy to understand / track
- Efficient and Scalable
- Easy to train with expert knowledge

Multilayer Perceptron

Overview



Linear Regression



Multilayer Perceptron
A.k.a.
Deep Learning

When To Use Deep Learning

You like magic (a.k.a. always)

When To Use Deep Learning

(computational) cost is no issue

When To Use Deep Learning

Boosting resume for next job

When to Use Deep Learning

- No idea of underlying data structure (brute force model)
- Seriously, no one cares how your model works.
- Seriously, you have a lot of compute power to train with
- High dimensional output space (this part of the image is a cat)

Use Case

Scenario

Fleet maintenance.

We have engine telemetry. Sensor readouts. Etc.

We want the savings in fleet maintenance to be greater than IoT implementation costs.

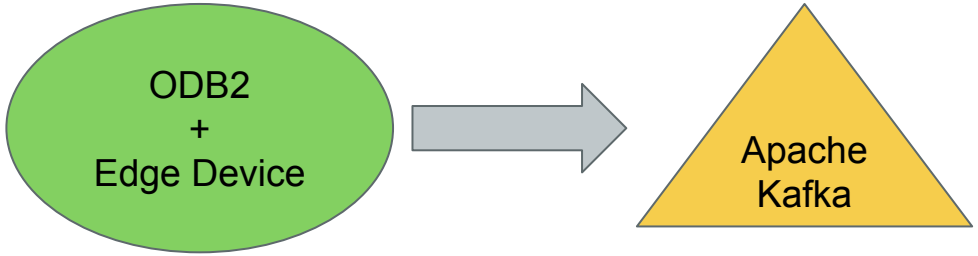
1. “Predict Engine Failure” (modern ECMs do this to a non trivial extent)
2. Recommended driver re training.

ODB2 Sensor

Reading:

- Throttle position sensor
- Hard Braking Sensor
- Engine Codes
- Coolant temp
- MAF Voltage
- Lat / Lng
- ...

Stack



Training (Bad Driver)

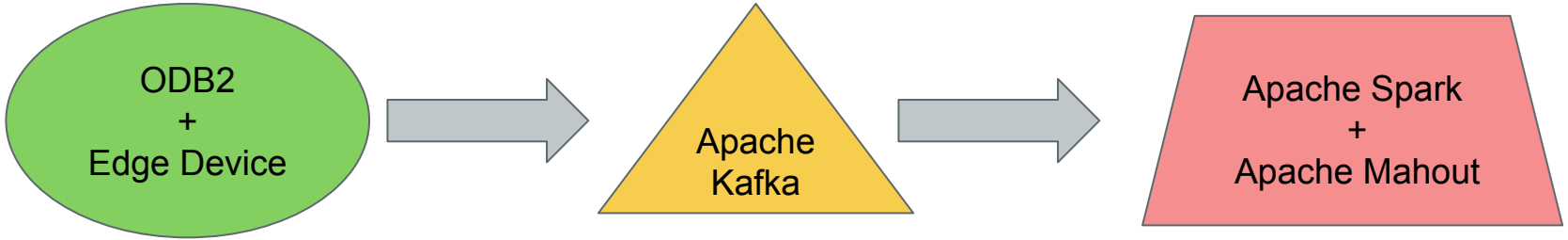
CCO Recommender
Rows (Users) = Driver
Primary Action = Service Required

Secondary “Actions”

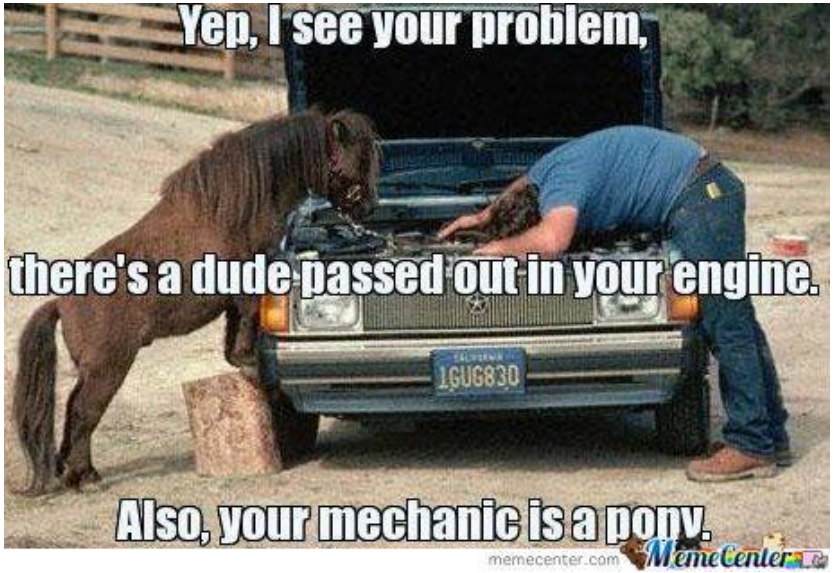
- Driver history of hard accel
- Driver hx of hard break
- Driver experience
- Etc

“Recommending Service based on Driver History” <- Calculate drivers who are most likely to cause service ~~incid~~ents based on their history.

Stack



Expert Training (Maintenance)

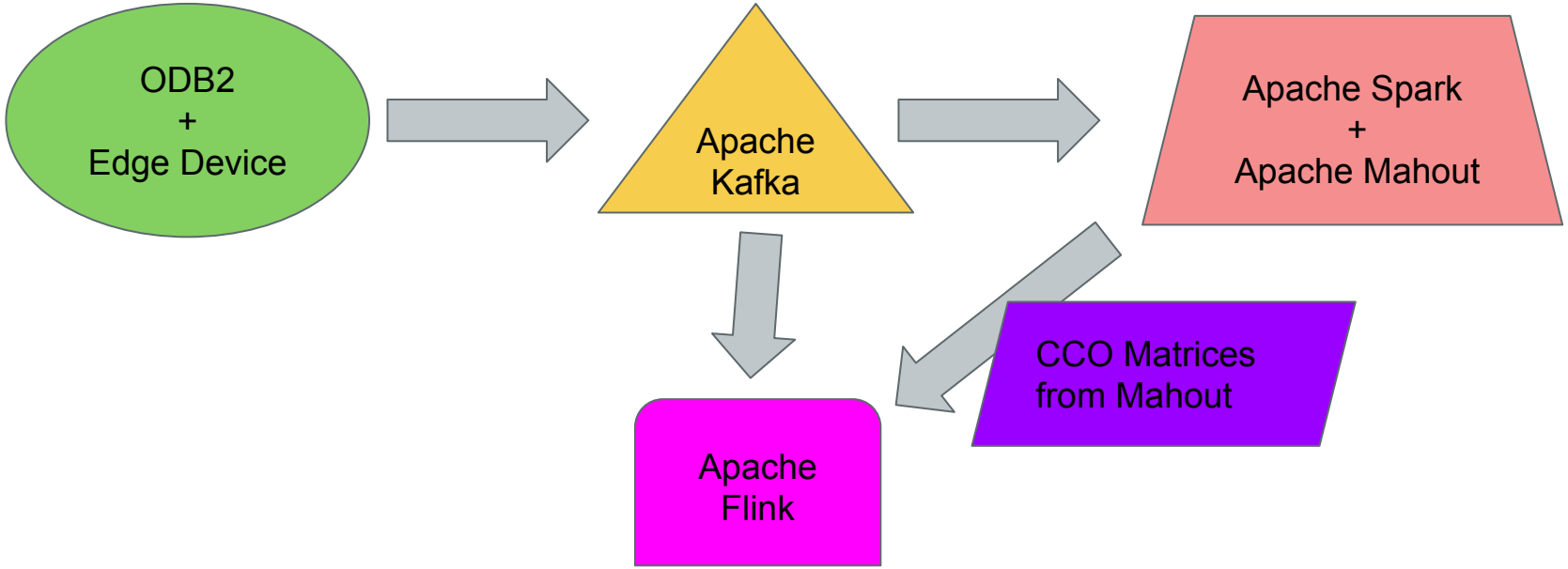


During initial period

- Engine Logs recorded
- Vehicle comes in for maintenance- “Expert” reviews logs and tags when vehicle “should” have come in, for what.

Scenario 1
Server Side

Stack


ODB2
+
Edge Device
Apache
Kafka
Apache Spark
+
Apache Mahout
CCO Matrices
from Mahout
Apache
Flink

Scenario 2
Edge Device Side

In Core Recommenders

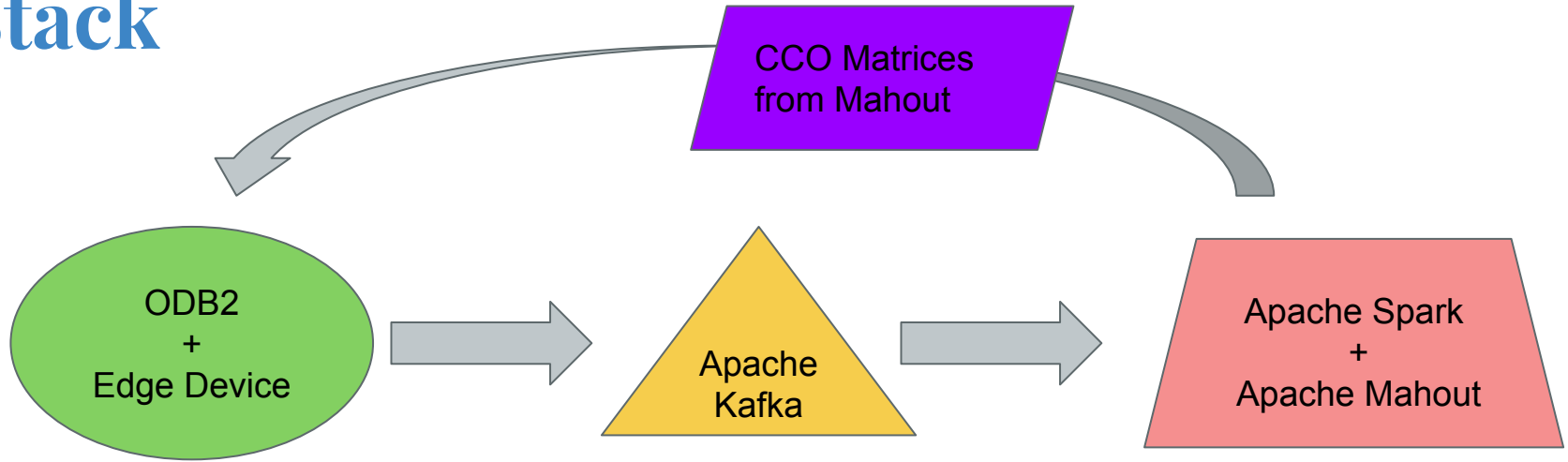
Simple app on in-vehicle Raspberry Pi streams data in from ODB2 Device

Recommender matrices are pushed out to all vehicles

Recommendations calculated at vehicle on Raspberry Pi (or other edge device)

Utilize Mahout “native solvers” for device architecture specific BLAS acceleration.

Stack


CCO Matrices
from Mahout
ODB2
+
Edge Device
Apache
Kafka
Apache Spark
+
Apache Mahout

Scenario 3
Micro Service Based

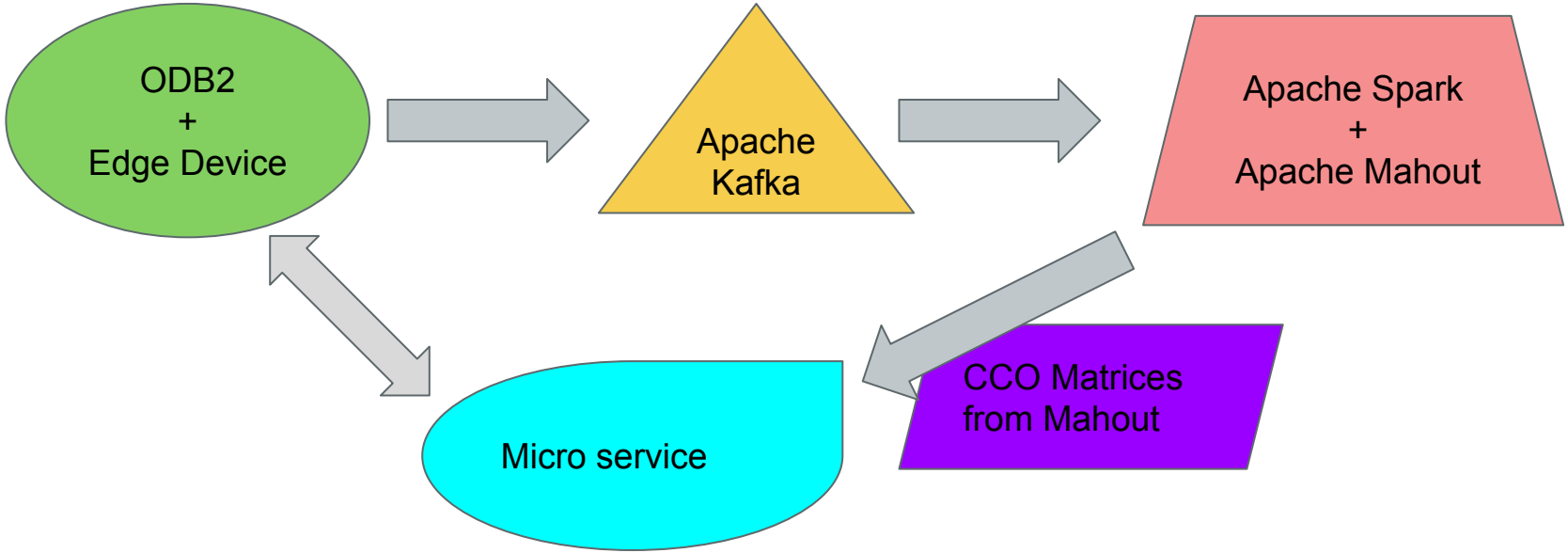
Modern Approach

Instead of calculating locally, REST call is made.

Better because less work at edge device (re: calculation, smaller edge devices)

Worse because fails if can’t make contact with server.

Stack


ODB2
+
Edge Device
Apache
Kafka
Apache Spark
+
Apache Mahout
Micro service
CCO Matrices
from Mahout

Questions